CLAIM AMENDMENTS

Claims pending:

- At time of the Office Action: Claims 1-40.
- After this Response: Claims 1-41.

Canceled claims: None.

Amended claims: 34 and 35.

New Claim: 41.

The listing of claims below will replace prior versions of claims in the application:

1. **(Previously Presented)** In a computer device that uses flash memory to store data, a method comprising:

maintaining one or more mapping data structures containing mappings of logical flash memory addresses to physical flash memory addresses, each mapping data structure having a predetermined capacity;

allocating additional mapping data structures as needed to provide capacity for additional mappings;

removing one or more additionally allocated mapping data structures if the capacity of mappings is not needed; and

maintaining a master data structure containing a pointer to each of the one or more mapping data structures, wherein the number of pointers changes according to the number of data structures.

2. **(Original)** The method as recited in Claim 1, further comprising adding pointers to the master data structure for the additionally allocated mapping data structures.

3. **(Original)** The method as recited in Claim 1, wherein the mapping data structures and master data structures are generated by a flash driver.

4. **(Previously Presented)** The method as recited in Claim 1, wherein the mapping data structures and master data structures are stored in a volatile memory device of the computer device.

5. **(Canceled)**

6. **(Previously Presented)** A system for tracking data in a flash medium, comprising:

a secondary data structure containing logical sector address to physical sector address mappings showing a relationship between logical sector addresses, requested by a file system, to physical sector addresses in which associated data is physically stored on the flash medium;

means for allocating a third data structure, if the secondary data structure becomes full, wherein the third data structure contains logical sector address to physical sector address mappings and for deallocating the third data structure in the event the secondary data structure is sufficient for mapping physical sector addresses containing data to logical sector addresses; and

a master data structure containing one or more pointers that point to the secondary data structure and the third data structure, if allocated, wherein the number of pointers changes as the third data structure is allocated and de-allocated.

7. **(Canceled)**

8. **(Previously Presented)** The system as recited in Claim 6, further comprising a flash media driver configured to determine how many physical sectors are contained on the flash medium.

9. **(Previously Presented)** The system as recited in Claim 6, wherein the means for allocating the third data structure is a flash driver configured to monitor how many logical sector address requests are issued by the file system to ensure there is enough data structure(s) allocated in addition to the secondary data structure.

10. **(Canceled)**

11. **(Previously Presented)** The system as recited in Claim 6, further comprising means for allocating a fourth data structure, if the second and third data structures are full.

12. **(Previously Presented)** The system as recited in Claim 6, wherein the data structures are stored in a volatile memory device.

13. **(Previously Presented)** A system, comprising:

a master data structure containing 1 to N pointers, wherein N is an integer greater than 1;

a secondary data structure containing mappings of logical sector addresses to physical sector addresses, the logical sector addresses contained in the secondary data structure being a portion of the maximum possible quantity of logical sector addresses that can be issued by the file system;

one or more additional data structures containing mappings of logical sector addresses to physical sector addresses, the one or more additional data structures being allocated when the portion of logical sector addresses contained in the secondary data structure is insufficient to store logical sector address write requests issued by the file system and deallocated if the portion of logical sector addresses contained in the secondary data structure becomes sufficient to store the logical sector address write requests issued by the file system; and

wherein the number of pointers in the master data structure changes as the one or more additional data structures are allocated and deallocated.

14. **(Canceled)**

15. **(Canceled)**

16. **(Canceled)**

17. **(Previously Presented)** The system as recited in Claim 13, further comprising a flash driver having a flash abstraction layer configured to monitor logical sector address requests by the file system and update the mappings of logical sector addresses to physical sector addresses.

18. **(Previously Presented)** The system as recited in Claim 13, wherein the master and secondary data structures are stored in a volatile memory device.

19. **(Previously Presented)** The system as recited in Claim 13, wherein the master and secondary data structures are stored in a random access device.

20. **(Previously Presented)** The system as recited in Claim 13, further comprising a flash driver configured to determine a size of a flash medium.

21. **(Previously Presented)** A computer device, comprising:

a flash driver configured to serve as an interface between a file system and the flash memory medium;

a master data structure containing enough pointers to match a maximum quantity of logical sector addresses to be issued by the file system;

a secondary data structure containing mappings of logical sector addresses to physical sector addresses, the logical sector addresses contained in the secondary data structure being a portion of the maximum possible quantity of logical sector addresses to be issued by the file system, wherein ones of the pointers from the master data structure point to specific mappings of logical sector address to physical sector addresses in the secondary data structure; and

one or more additional data structures containing mappings of logical sector addresses to physical sector addresses, allocated by the flash driver when the portion of logical sector addresses contained in the secondary data structure is insufficient to store logical sector address write requests issued by the file system and deallocated by the flash driver if the portion of logical sector addresses contained in the secondary data structure becomes sufficient to store the logical sector address write requests issued by the file system, wherein others of the pointers from the master data structure point to specific mappings of logical sector address to physical sector addresses in the one or more additional data structures, such that the number of pointers in the master data structure pointing to the second and additional data structures varies according to the number of data structures.

22. **(Canceled)**

23. **(Canceled)**

24. **(Canceled)**

25. **(Original)** The computer device as recited in Claim 21, wherein the flash driver comprises a flash abstraction layer configured to monitor logical sector address requests by the file system and update the mappings of logical sector addresses to physical sector addresses.

26. **(Previously Presented)** The computer device as recited in Claim 21, wherein the master and secondary data structures are stored in a volatile memory device.

27. **(Previously Presented)** The computer device as recited in Claim 21, wherein the master and secondary data structures are stored in a random access device.

28. **(Previously Presented)** The computer device as recited in Claim 21 wherein the computer device is a portable data storage and processing device.

29. **(Previously Presented)** In a computer device that uses flash memory to store data, a method comprising:

generating a master data structure containing a plurality of pointers;

allocating a secondary data structure used to map logical sector addresses to physical sector addresses, wherein the secondary data structure is limited in size;

enabling one of the plurality of pointers to point to the secondary data structure;

allocating a third data structure used to map logical sector addresses to physical sector addresses, if the secondary data structure fills-up, and deallocating the third data structure if the second data structure is no longer filled up; and

enabling one of the plurality of pointers to point to the third data structure, if allocated, such that the number of pointers pointing to data structures changes as the third data structure is allocated and deallocated.

30. **(Original)** The method as recited in Claim 29, wherein the logical sector addresses are issued by a file system and the physical sector addresses indicate where data associated with the logical sector addresses is physically stored on the flash medium.

31. **(Original)** The method as recited in Claim 29, further comprising ascertaining a quantity of physical sectors on the flash medium prior to generating the secondary data structure.

32. **(Original)** The method as recited in Claim 29, further comprising ascertaining a quantity of physical sectors on the flash medium prior to generating the secondary data structure and determining an address bit length for the pointers in relation to the quantity of physical sectors ascertained.

33. **(Original)** The method as recited in Claim 29, wherein the secondary data structure is $b*k$ bytes in size, wherein k is a number of physical sector addresses contained in the data structure and b is a number of bytes required to store each physical sector address.

34. **(Currently Amended)** One or more computer-readable media comprising computer-executable instructions that, when executed on the computer device, perform the method as recited in Claim 29.

35. **(Currently Amended)** The method as recited in Claim 29, wherein the computer device is a portable processing device.

36. **(Original)** The method as recited in Claim 29, wherein the method is performed by a flash driver in conjunction with the file system of the computer device.

37. **(Previously Presented)** The method as recited in Claim 29, wherein the data structures are stored in a volatile memory portion of the computer device.

38. **(Original)** The method as recited in Claim 29, wherein the secondary data structure fills-up when the logical sector addresses exceed the limited size of the secondary data structure.

39. **(Previously Presented)** A computer-readable medium for a Flash driver, comprising computer-executable instructions that, when executed, direct the Flash driver to:

generate a master data structure containing one or more pointers;

allocate a secondary data structure used to map logical sector addresses to physical sector addresses, wherein the logical sector addresses are issued by a file system and the physical sector addresses indicate where data associated with the logical sector addresses is physically stored on the flash medium;

allocate a third data structure used to map logical sector addresses to physical sector addresses, if the secondary data structure fills-up, and deallocate the third data structure if the second data structure is no longer filled up; and

enable pointers from the master data structure to point to the second and third data structures, wherein the number of pointers changes as the third data structure is allocated and deallocated.

40. **(Original)** The computer-readable medium as recited in Claim 39, further comprising computer-executable instructions that, when executed, direct the Flash driver to allocate one or more additional data structures in the event that the third data structure fills-up.

41. **(New)** A computer-readable medium for a Flash driver, comprising computer-executable instructions that, when executed, direct the Flash driver to:

maintain one or more mapping data structures containing mappings of logical flash memory addresses to physical flash memory addresses, each mapping data structure having a predetermined capacity;

allocate and deallocate additional mapping data structures as capacity for additional mappings changes; and

maintain a master data structure containing at least one pointer to each of the mapping data structures, wherein the number of pointers changes according to the number of data structures.